REMARKS

Claims 1-55 are pending in the current application. Applicants have amended independent claims 1 and 49 to more clearly recite features of Applicants' invention. Claims 2-48 and 50-55 have been amended to correct minor informalities. No claims are cancelled or are newly presented. No new matter has been introduced by the claim amendments provided herein.

Support for "importing at least one particular test case" in claim 1 is provided, *inter alia*, in the specification at paragraph [0032]. Support for "using semantic analysis to decompose the at least one test case into a platform-independent abstract representation which comprises at least three components, and wherein the at least three components include at least one application state, at least one external interaction sequence, and input data" in claim 1 is provided, *inter alia*, in the specification at paragraph [0036] and in Figs. 1 and 9, and in the Abstract. Further amendments to claim 1 are provided for clarity and to correct minor informalities. Support for amendments to independent claim 49 as provided herein are found at least in these same portions of the specification.

Objections to the Claims Should Be Withdrawn:

The examiner objects to claims 1 and 49 for minor informalities. Applicants have amended claims 1 and 49, as provided in the current claim listing herein, to address the Examiner's objections.

Rejections under 35 U.S.C. §103 Should Be Withdrawn

Independent Claim 1

Claims 1-55 stand rejected under §103(a) as being unpatentable over U.S. Patent No. 7,313,564 to Melamed et al. ("Melamed") in view of the WinRunner 7.0 Tutorial, Mercury Interactive Corp. 2000 ("WinRunner"). This ground of rejection is respectively traversed.

Independent claim 1 as amended herein recites a method for generating test scripts which includes, inter alia, **using semantic analysis to decompose the at least one test case into a platform-independent abstract representation which comprises at least three components, and wherein the at least three components include at least one application state, at least one external interaction sequence, and input data**. The Examiner acknowledges that Melamed does not explicitly disclose this limitation, and alleges that WinRunner does. (See Office Action, page 5).

Specifically, the Examiner alleges that the GUI Map described in WinRunner corresponds to such an abstract representation. (Id.) Applicants respectfully disagree. The GUI Map described in WinRunner identifies certain properties of GUI elements in the software application being tested. (See WinRunner, pp. 22-26). A GUI Map can be created for each test generated by a user of the WinRunner program. (See WinRunner, pp. 27-29). However, the GUI Map is **not** an abstract representation of a test case which includes at least one application state, at least one external interaction sequence, and input data as recited in claim 1. The GUI Map merely describes properties and status of individual GUI elements which are present in a tested application.

The Examiner further alleges that the abstract representation of a test case recited in claim 1 corresponds to various recording modes which can be used in the WinRunner program, such as Context Sensitive Mode and Analog Mode, and which are described on pp. 36-50 of WinRunner. (See Office Action, page 6). Applicants respectfully disagree. As described on page 35 of WinRunner, the recording modes merely allow generation of automated test scripts by recording specific user operations and ***generating statements in TSL, Mercury Interactive's Test Script Language***. (See WinRunner, p. 35). Thus, WinRunner does **not** import a test case and decompose it into an abstract representation using semantic analysis, as recited in claim 1. Instead, the recording modes in WinRunner allow a user to ***generate*** a specific test script by merely recording interactions of the user with the application being tested. Further, the test script in WinRunner is generated in a ***specific*** scripting language (TSL), and is **not** provided as an abstract representation which includes application states, external interaction sequences, and input data, as recited in claim 1. Also, application states, external interaction sequences (test steps) and input data are **combined** within the

scripts generated and used by WinRunner. It is difficult or impossible to modify or reuse such integrated test cases. In contrast, claim 1 recites a method which uses abstract representations of test cases in which these components of a test case are provided **separately**.

The Examiner fails to identify any **semantic analysis** which is used to **decompose a provided test case into an abstract representation**, as recited in independent claim 1. An example of a semantic analysis approach to decomposing an imported test case is shown, e.g., in FIG. 7 of the present application and referenced in paragraph [0037]. Neither Melamed nor WinRunner describe such a **semantic analysis** procedure for **decomposing a provided test case into a *platform-independent abstract representation***. Instead, Melamed describes providing test case files to a host machine, where the test case files identify a particular automation tool (e.g., WinRunner) associated with the test case. (See, e.g., Melamed, col. 11:50-12:14). The host machine then parses the test case to generate a specific test script, and uses the parsed test script with the separate GUI environment file associated with the specific automation tool to execute the test script. (See Melamed, col. 12:54-64). The system and method described in Melamed parse a provided test case into a **platform-specific** test script which further requires an additional **platform-specific** GUI environment file to be executed by the host machine. WinRunner describes the generation of test scripts in its **platform-specific** TSL language by recording specific user interactions. Accordingly, neither WinRunner nor Melamed discloses any **decomposition of a test case into a platform-independent abstract representation using semantic analysis**, as recited in independent claim 1.

The Examiner further alleges that Melamed generates rule-based test cases based on an abstract representation. (See Office Action, page 5). Applicants respectfully disagree. Melamed does not describe or suggest such a **platform-independent abstract representation** which is obtained by decomposing a test case using semantic analysis, and which includes application states, external interaction sequences, and input data, as recited in amended claim 1. Instead, the portions of Melamed relied on for these recitations (col. 13:34-14:49, FIGS. 1-8 and related text, FIG. 18, and col. 12:7-64) refer to test cases being created using the **platform-specific**

Integrated Test Case Authoring Tool, and sent to a particular host machine for parsing into **platform-specific** test scripts and subsequent execution. As illustrated in FIG. 18 of Melamed, these test cases 182 are **not** rules-based test cases generated from an abstract representation, but are specific test cases which are encoded into a convenient file format (e.g., XML or flat file) in an application server, and sent to a host machine 184, where they are parsed into test scripts and executed using a GUI File created by an automation package, which is also sent from the application server to the host machine. (See Melamed, col. 12:54-64 and FIG. 18).

For at least these reasons, Applicants respectfully assert that the alleged combination of Melamed with WinRunner fails to disclose all of the features recited in independent claim 1 as amended herein.

Independent Claim 49

Independent claim 49, as amended herein, recites a computer system configured to perform the method recited in independent claim 1. Accordingly, for at least the reasons provided above with respect to claim 1, Applicants respectfully submit that the alleged combination of Melamed with WinRunner also fails to disclose all of the features recited in independent claim 49 as amended herein.

Dependent Claims 2-48 and 50-55

Claims 2-48 depend directly or indirectly from independent claim 1, and claims 50-55 depend directly or indirectly from independent claim 49. For at least the reasons provided above with respect to independent claims 1 and 49, Applicants respectfully submit that the alleged combination of Melamed with WinRunner also fails to disclose all of the features recited in dependent claims 2-48 and 50-55.

Further, with respect to dependent claim 11, the Examiner alleges that Melamed discloses validation of rule-based test cases. As an initial matter, Melamed does not teach or suggest any **platform-independent abstract representation of a test case**, as discussed above with respect to claim 1, and therefore fails to disclose **generating a rule-based test case based on such an abstract representation**. Further, Melamed fails to teach or sugest any **validation of a rule-based test case** as recited in claim 11.

The test case manipulation in Melamed relied on by the Examiner describes a mapping of test cases to requirements/sub-requirements. (See Melamed, col. 9:17-19, and FIG. 5). This mapping relates to organizing and managing both requirements and test cases together. In contrast, claim 11 recites **validating rule-based test cases** themselves, which allows automatic determination of which of such test cases may be valid or broken without actually running the test cases or manually verifying them.

Dependent claims 12-30 recite further features of such a validation of rule-based test cases which Applicants again assert are not disclosed by Melamed for at least the reasons provided above with respect to claim 11.

The Examiner further alleges that Melamed discloses **providing rules for selection of components of test case definitions, external interaction sequences and input data; and further providing rules for data driven test case generation**, as recited in dependent claim 33. Applicants respectfully disagree. The portion of Melamed relied on by the Examiner for these recitations describes identification of individual GUI elements in an application to be tested, and manual selection of preset testing procedures for the selected object which may be added to a test case. (See Melamed, col. 15:41-col. 16:21). This manual selection of **specific testing procedures** to be run on **specific GUI objects** selected by a user, as described in Melamed, provides editing capabilities for a test case. However, Melamed does not teach or suggest **providing *rules* for selection of components of test case definitions, external interaction sequences and input data**, nor **providing *rules* for data driven test case generation**, as recited in dependent claim 33. Instead, Melamed merely describes certain user interface capabilities which allow manual editing of test cases by adding specific test procedures to selected GUI objects, and fails to disclose any **rules** for selection of components or for test case generation.

Dependent claims 34-43 recite further features of such rules for selection and test case generation, which Applicants again assert are not disclosed by Melamed for at least the reasons provided above with respect to claim 33.

CONCLUSION

It is submitted that the pending claims 1-55 of the present application as amended and presented herein are in form for allowance, and such action is respectfully requested. The Commissioner is authorized to charge any additional fees which may be required, including petition fees and extension of time fees, to Deposit Account No. 07-1700 (Docket No. SYM-0004).

Respectfully submitted,
GOODWIN PROCTER LLP

Date: 6.26.08

Paul Davis, Reg. No. 29,294

135 Commonwealth Drive
Menlo Park, CA 94025
Tel: 650/752-3106
Customer No. 77845